

JUL 10 1979

L271:515

The Development and Uses of Table Producing Language

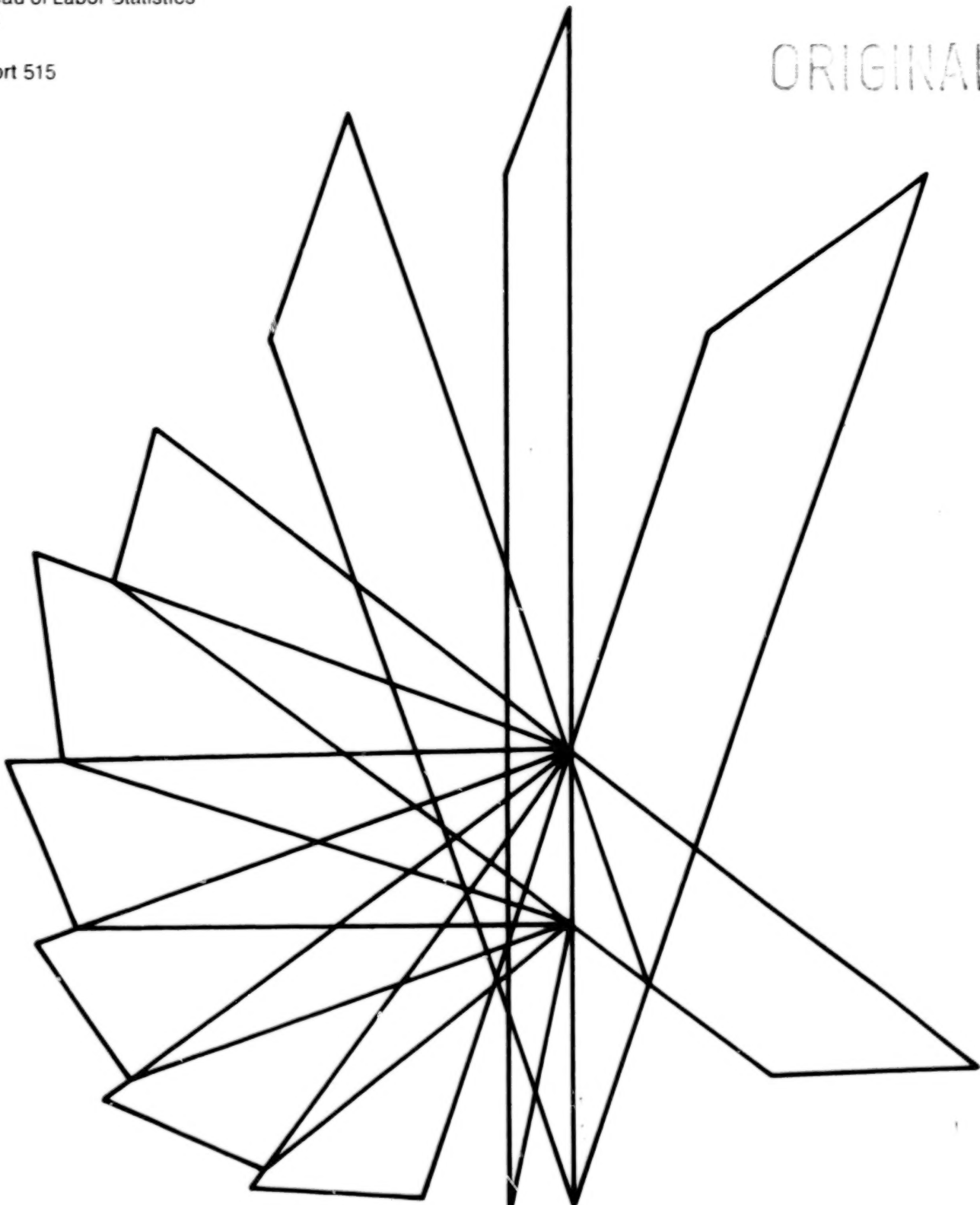


COMPLETED

U. S. Department of Labor
Bureau of Labor Statistics
1978

Report 515

ORIGINAL



U.S. Department of Labor
Ray Marshall, Secretary
Bureau of Labor Statistics
Julius Shiskin, Commissioner

1978



Preface

This report describes the development and uses of a Bureau of Labor Statistics (BLS) computer program known as Table Producing Language (TPL). Through a unique system of retrieving and processing information from computer storage, this program reduces the time between data collection and presentation of results in statistical tables. A User Manual is available for anyone interested in studying TPL or in acquiring a copy of the complete system.

TPL was developed in the Office of Systems and Standards by the Division of General Systems, under the supervision of Peter B. Stevens. Design and development were headed by Richard W. Hedding, who was assisted by Pamela L. Weeks, Nancy J. Byrd, Victor G. Stotland, Stephen B. Levenson, John D. Sinks, Roxana B. Kamen, Stephen E. Weiss, Eugene C. McKay, Walter L. Taylor, Hania M. Schwedt, and Jane E. Powers. Robert J. McIntire and Kenneth Buckley developed the interface procedure between TPL and routines for statistical analysis.

This report was prepared by Rudolph C. Mendelssohn, Assistant Commissioner, Office of Systems and Standards, Bureau of Labor Statistics.

Material in this publication is in the public domain and may be reproduced without permission of the Federal Government. Please credit the Bureau of Labor Statistics and cite the name and number of the publication.

BLANK PAGE

Contents

	<i>Page</i>
Introduction	1
Needs and goals	1
Building a system	1
The road to TPL	3
How TPL works	4
Methods and components	6
Variables side by side	6
Nesting	6
Wafers	7
The table statement	7
Rules for the nest and concatenate operators	7
Complex applications	7
Other TPL statements	8
The codebook	11
Functions	11
Practical uses of TPL	13
Technical considerations	16
Special problems at national statistical agencies	16
TPL compared	16

Introduction

Table Producing Language (TPL) is a computer language system designed by the BLS Office of Systems and Standards to select, restructure, cross-tabulate, and display Bureau of Labor Statistics (BLS) data. The system was designed to reduce the Bureau's need for special computer programs that produce cross-tabulations, expand its tablemaking capability, and broadly speaking, improve production schedules by reducing time between collecting survey data and viewing the results in tabular form.

An important activity of the Bureau is the production of statistical tables for economic analysis and for publication in periodicals and bulletins. The Bureau is an important publishing house in the U.S. Government and statistical tables are the bulk of its printed output. Although the importance of statistical tables has been fairly clear to all BLS employees, the Bureau spent little time trying to understand table anatomy and construction. Questions such as the following rarely were asked by the Bureau's own analysts: "What is a statistical table? What parts does it have? How are the pieces put together? Are there rules?" And so on.

Needs and goals. Tables usually had been generated from micro data by creation of a new computer program for each new need. As a result, computer analysts and programmers repeatedly were assigned to plan, design, test, and debug special purpose tabulation programs. Although these programs supported the Bureau's statistical production processes, relatively few were designed to meet publication standards. Consequently, additional resources, for example, typing or typesetting, often were required to convert computer output into an acceptable form for reproduction, a time-consuming and expensive process. Equally important, the high cost of creating tabulation programs inhibited their availability for research. Hence, both BLS production and economic analysis suffered.

During the first half of 1971, the BLS undertook a study of the range of tables it produced to see if they shared common characteristics that could be dealt with in a single computer system.^{1 2} The study revealed that these tables fall into three broad classes: Those published in the Bureau's bulletins and reports, work tables used in the preliminary stage of preparing data, and a third class more difficult to observe. BLS professional personnel rely heavily on the Bureau's massive data files for research. The form of the tabulations from

these files is not predictable because the analyst typically engages in an interactive process, that is, the study of one table leads to new questions which require different tables which generate new questions, and so on until the analyst is satisfied.

The study revealed one dominant fact: There was no agreement on how to describe tabulation methods and table formats among the computer systems staff, economists, statisticians, demographers, and other social scientists throughout the Bureau. Terms like variable, data element, data item, and field often were interchanged, depending on the context or the user's background. Simple words like row, line, column, table, summary, and cross-tabulation had varied interpretations. Nor did a look at other tabulation systems help. To improve communication among BLS social scientists, computer science-professionals, and the computer itself, the BLS decided to standardize the language used to describe its tables.³ The machine then would take the description and develop procedures to manufacture the product.

Briefly, there were four goals for the system:

1. It should produce most, if not all, of the Bureau's statistical tables.
2. It should be driven by a language that did not require the user to be competent in the computer science discipline.
3. It should be flexible and adaptable to changing needs for new tables and formats.
4. It should lead the way to composition of tables for publication.

Building a system. An analysis of the study findings indicated that in building the standardized language to describe tables, parts of the table had to be named and an unambiguous syntax devised. After the language was completed, a computer system could be designed to create the tables from user descriptions.⁴

Accordingly, the TPL idea began to jell, and two major components of the system were defined: A codebook and the language itself. The codebook, a one-time detailed description of the file, must be prepared by someone familiar with the file's physical characteristics and content, including all portions of any record which may contribute to tabular output. Variables must be named, and data lengths and locations specified. Once the codebook is available, many TPL users can reference a file without knowing the details of its

organization. A computer program converts the code-book to a special internal format which the second major component, the TPL, may use to generate the tables needed.

A set of TPL statements instructs the computer to select and tabulate the data in a specified format. The user specifies the variables and gives instruction for the table stub and head. The TPL system can (1) place variables side by side, (2) allow levels of subdivision (variables within variables), and (3) designate separate,

two-dimensional grids or the repetition of the grid for additional variables.

The system also can calculate averages, medians, minima, maxima, quantiles, and relative time. It also can create new variables from existing variables, calculate additional data after tables have been compiled, deal with subsets of information in the data file, and group, delete, or reorder values of existing variables. Many tables can be produced in a single run. Further, the user can arrange output in any sequence.

The Road to TPL

In constructing the system, the first step was to study what others had done, particularly, the work of other national statistical agencies. A United Nations questionnaire sent to national statistical agencies in Europe, Australia, and North America in 1972 disclosed nearly 50 systems that produced tables. So much activity certainly demonstrates that most statistical offices regard some degree of generalization desirable and possible. But two questions are raised:

1. Why so many different systems?
2. Why not use one of these in BLS rather than develop a new one?

Differences in computers and in data file structures limit the interchangeability of programs. However, this does not explain why some organizations have three or four different systems and why BLS decided to develop its own. Almost every system examined was capable of doing something useful, but no system came close to meeting all the Bureau's requirements, individually or collectively.

To some extent, the Bureau was guided by industry experience in developing generalized "sort" routines. In the early days of computers, people used to write sort programs from scratch. Because the process could be generalized, computer manufacturers began to include sort programs as part of their standard software. These programs are used so universally that the phrase "write a sort" has come to mean "fill out the control cards for the general sort program." Almost no one writes sort programs anymore.

Sorting and cross-tabulation are closely related, for both processes order data into a sequence based on their characteristics. Cross-tabulation goes beyond sequencing to summarization of like data elements; only the summaries remain at the end. Sorting and cross-tabulation involve significant programming complexity and in both cases the most complex algorithms do not

change from job to job.

The choice of programming languages for generalized systems has been of great interest in the computer industry. The Bureau rejected assembly or machine language as too costly and time consuming. Similarly, FORTRAN and COBOL lacked the language features necessary for efficient compiler and statistical system construction. PL/1 contains all of the necessary language elements but it is large and complicated.

The first problem was to define a language that would describe tabulations; the second problem was to construct a compiler that would include these difficult algorithms. Both solutions were facilitated by using the XPL Compiler Generator System.⁸ The XPL system, which includes a dialect of PL/1, is designed for compiler construction but has great power for language construction as well.

The XPL language, which is much simpler and smaller than PL/1, contains all the features necessary for compiler writing. Its facilities for character string manipulation are quite good and its operations very fast. Of greatest importance to BLS was the generalized grammar analysis and parsing (comprehension) program. The analysis program accepts as input the description of a grammar in BNF notation,⁹ tests the grammar for correctness, and generates the tables necessary to drive a generalized scanning and parsing program. It accepts grammars which can be parsed by looking ahead one symbol in the input text. The program's tests for correct form and unambiguity are quite rigorous and its operations may be trusted.

By using XPL, BLS systems personnel could concentrate on language design and execution rather than become deeply involved in the complexities of parsing algorithms, source language scanning, and compiler organization. The savings in energy, time, and cost have been very large.

How TPL Works

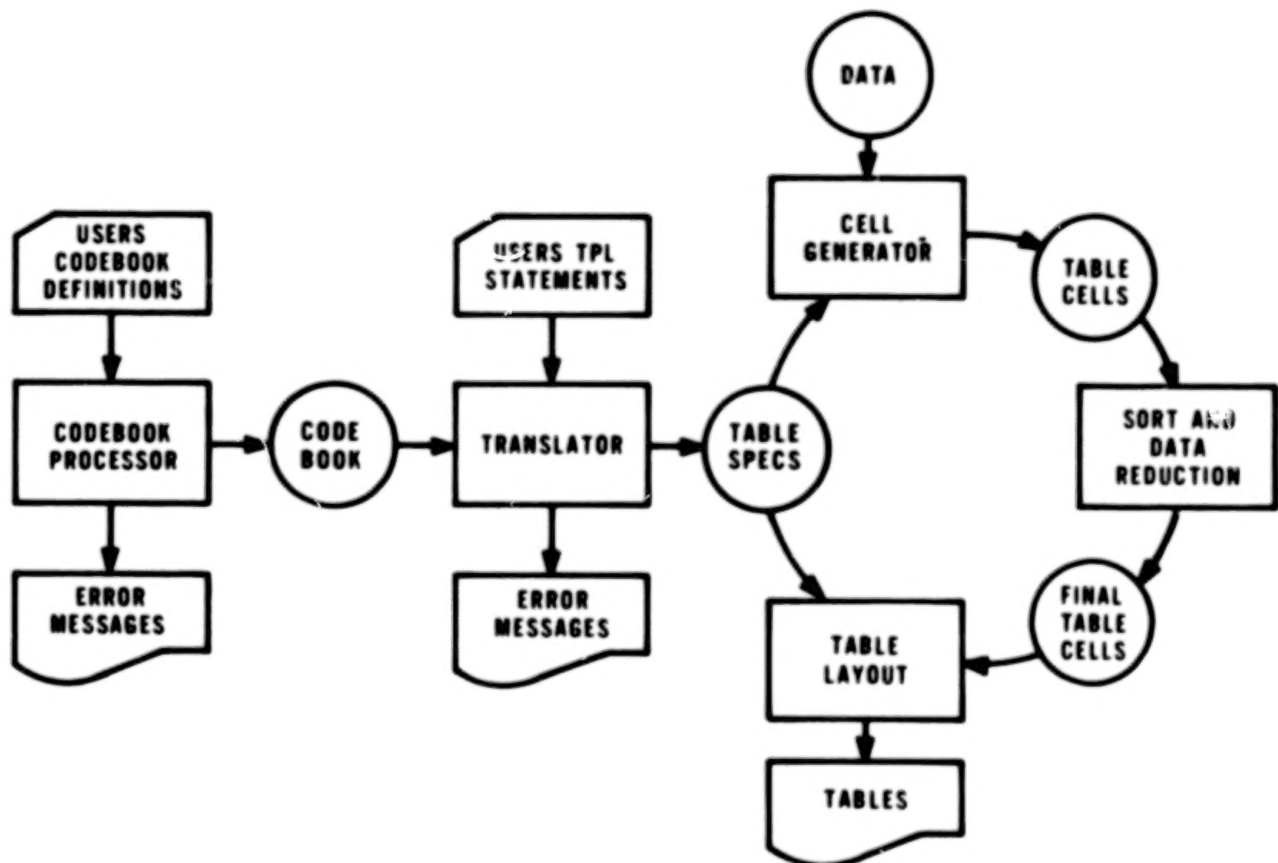
The key to the system lies in the way data are processed. A similar approach was used earlier by BLS⁷ and by the Australian Bureau of Statistics.⁸ In TPL, identification and retrieval of the data item are separate from compilation and table preparation. Figure 1 shows how this approach works.

The heart of the system is a routine called "Translator." Its function is to accept user-written TPL statements that cite the variables needed and describe how they will appear in the tables. The user request is compared with relevant data in the codebook file. The translator extracts and converts to table specifications information about record type, field length, location of variable values in the data file, and names for printing

in table headings, columns, and stub. These specifications are delivered to the cell generator program which sets up all the data-dependent information for computer processing. As the cell generator searches a tape or disk file for microdata to tabulate summary statistics, it sets up records for each datum used (one record for each instance in which the number is used) and tags the records to reflect their ultimate use and disposition of computed results.

When retrieved, each datum is deposited on its own unique and completely independent record. That record includes a space to deposit the retrieved datum, a code to identify the statistical table to which the datum

Figure 1: The Table Producing Language System Flow Diagram



belongs, coordinate numbers to show the line and column of the figure in its table, and the wafer to which it belongs. These codes will then control the record.

After all data have been retrieved (table cells, figure 1), the system sorts the records according to the processing codes. A sort on table number brings related work together for each table. Keying on line number, column number, and so forth organizes the data within the table. Data reduction then takes place and the statistics for each table are derived. The process, in fact, is more complex than described here. Summarization will take place during the cell generator phase, as well as during the sort step, as soon as the system recognizes that the needed data are at hand. This approach affords machine efficiency.

Each datum record is totally disengaged from all related data or records and floats about freely, guided only by the codes which are attached to it. Because limitations are not imposed by the sequence of the input data, the user can arrange the output in any sequence. Many tables can be run at one time and their structures can differ from each other without limit.

The last step forms the table. Statistics delivered to a table layout routine are joined by additional translator information, such as alphabetic content of headings, stubs and table titles, guidelines for setting column widths, and special features such as hyphenation, centering of titles and headings, and other aids that make the table sensible and easy to read.

One important, and perhaps unique, feature of the BLS system can be underscored at this point. TPL uses the machine central processing unit (CPU) primarily for operations, rather than for storage; that is, selecting, computing, and processing. Retrieved data elements which pile up to CPU core storage capacity are shunted to secondary storage devices (tape or disk) where they are held until sorted according to need. This design approach reveals why there is in one run no theoretical limit (1) to the size of the table that can be processed, (2) to the levels of cross-tabulation, and (3) to the number of variables processed. There is, of course, a physical limit to the amount of main memory, secondary disk memory, and tape storage that can be lined up and tied together for a job.

Methods and Components

In construction of the TPL language, BLS requirements were reduced to one of the most elementary concepts of a table, namely a tabulation of two variables such as age and income. Testing this simple model with BLS staff was not easy because of the communications gap. A minor case will illustrate the procedures used to develop the system. Some BLS analysts would specify—Age by Income; and others—Income by Age. Neither recognized the need for conventions unless a question was raised about which variable is to be the stub and which is to be the heading.

Although the numbers in table 1 would be no different if the heading and stub were transposed, the values and the table appearance (often an important consideration) would be reversed. In practice, some BLS economists and statisticians resorted to drawings, not always successfully, because the arithmetic to tabulate the table could not be pictured.

Table 1: Age, Income

AGE	INCOME		
	1	2	3
1			
2			
3			

Variables side by side. The systems personnel found that setting variables side by side controlled the appearance of a table. A modification of the two variables, Age and Income, appears in table 2.

Although the same two variables are dealt with, values in this table would be quite different from those in table 1 where items classified by two criteria have been counted. In table 2 all occurrences of each of the two variables have been summed separately; that is, the number of persons in Age 1, the number of persons in Age 2, in Income 1, etc. Moreover, the results are not the product of cross tabulation. (Perhaps "tally" or "summary" would be more descriptive.) Misunderstandings about the arithmetic process also arose when nomenclature about appearance was ambiguous.

Table 2: Age THEN Income

AGE			INCOME		
1	2	3	1	2	3

Nesting. There were other misunderstandings in the two-variable example—Age, Income. Some analysts perceived the table as shown in table 3.

Table 3: Age BY Income

AGE 1			AGE 2			AGE 3		
INCOME			INCOME			INCOME		
1	2	3	1	2	3	1	2	3

This construction, which derives from the Cartesian notion of nesting, may be extended into nests within nests, etc. Many Bureau tables reflect this simple cross-tabulation model in complex, increasing levels of nesting (table 4). Thus, having already added variables side-by-side, nesting (levels of subdivision) had to be included as a basic condition that shaped a statistical table.

When a third variable was added, the range of table formats increased markedly, as did the consequent communication confusion as illustrated by Table 4: Sex BY Age BY Income, which can be organized in a dramatically different way (Table same 4) without the slightest change of meaning.

Table 4: Sex BY Age BY Income

MALE									FEMALE								
AGE 1			AGE 2			AGE 3			AGE 1			AGE 2			AGE 3		
INCOME			INCOME			INCOME			INCOME			INCOME			INCOME		
1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3

Table same 4: Sex, Age, Income

FEMALE	INCOME		
	1	2	3
MALE	INCOME		
	1	2	3
AGE 1			
AGE 2			
AGE 3			

As in table 1, each grid can have its stub and heading transposed and the values moved accordingly without having its substance changed although the appearance

differs. Because a request for a table of sex, age, and income can be met in several radically different ways, statisticians and colleagues in computer sciences found their conversations confusing.

Wafers. Another terminological gap was the absence of a word for the concept of a repeated table. In the next illustration, table 5 is repeated for regions. The word "wafer" designates separate, two-dimensional grids. Thus, the simple grid model had to be augmented repeatedly to include additional, irreducible concepts. Altogether, four fundamental forms were found: (1) a simple grid as in table 1 with variables in the heading and stub; (2) the form as in table 2 with variables placed side by side; (3) the grid with nesting as in table 3; and (4) the grid repeated as wafers, as in table 5.

Table 5: Region, Age, Income

REGION C	INCOME				
	1	2	3		
REGION B	INCOME				
	1	2	3		
REGION A	INCOME				
	1	2	3		
AGE 1					
AGE 2					
AGE 3					

The table statement. Because BLS tables were formed from a few basic building blocks, formal statements were constructed that would clarify table specifications.

The first of these, called the TABLE statement, defines the grid by naming the variables in the table stub and headings. When needed, it includes an expression for the repetition of the grid into wafers. Wafers are always cited before the stub and heading expressions. Thus, an analyst asking for a cross-tabulation like table 5 would describe the table wanted as having region for the wafers, age for the stubs, and income for the column headings.

The general form of the TABLE statement is:

TABLE name: Ew, Es, Eh;

where "name" identifies the table being specified and Ew, Es, and Eh are, respectively, the wafer, stub, and heading expressions, each separated from its neighbor by a comma. Thus, the statement for table 5 is:

TABLE 5: Region, Age, Income;

as illustrated in the table heading. When only one grid is wanted, the wafer expression is not used, e.g., TABLE name: Es, Eh;

Rules for the nest and concatenate operators. In addition to the formal statement framing the table, a discipline was established for nesting and for placing variables side by side within the table.

Cartesian nesting within a table is indicated by the capitalized word BY between variables when the second variable of a pair is to be nested within the first variable. As for example, in table 3:

Age BY Income

When one variable is placed next to another in a table, the capitalized word THEN should be used, as in table 2:

Age THEN Income

This is called the concatenation rule.

Nest and concatenation rules are used in a variety of combinations, seemingly with any number of variables in many BLS tables. For example, one can add to table 2: Age THEN Income, a requirement for having sex nested within income, as in table 6.

Table 6: Age THEN Income BY Sex;

AGE			INCOME 1		INCOME 2		INCOME 3	
1	2	3	MALE	FEMALE	MALE	FEMALE	MALE	FEMALE

Two more headings using the same three variables but reversing the operators indicate the variety of table structures that can be defined using only three variables and combinations of nesting and concatenation. Compare the headings of tables 6 and 7. Table 8 illustrates the nesting of sex within both age and income.

Table 7: Age BY Income THEN Sex;

AGE 1			AGE 2			AGE 3			SEX	
INCOME	INCOME	INCOME	INCOME	INCOME	INCOME	INCOME	INCOME	INCOME	MALE	FEMALE
1	2	3	1	2	3	1	2	3		

Table 8: Age BY Sex THEN Income BY Sex;

AGE 1		AGE 2		AGE 3		INCOME 1		INCOME 2		INCOME 3	
M	F	M	F	M	F	M	F	M	F	M	F

Complex applications. These powerful rules can be combined in so many ways that specifying complex tables could become awkward and cumbersome as in table 9. Because of these complexities, a way had to be found to make the combinations concise and clear in one unambiguous statement. A mathematical approach

lation of averages, and other, similar results for posting on the table. One is called the COMPUTE statement and the other, POST COMPUTE.

COMPUTE allows any calculation using the arithmetic operators of addition, subtraction, multiplication and division. For example:

COMPUTE A = (Interest + 3)/(Gross - Interest) * 5;

With the COMPUTE statement, it is possible to create values for new variables by combining values of other variables and constants. (Weighting would be an example of combining values of variables with constants.) These new variables, depending on the "condition" or value of other variables, can also be derived in more than one way.

The POST COMPUTE statement is used to calculate additional values after tabulation but before tables have been displayed. For example, suppose we require a table for three regions showing a column of total income followed by a column of the number of persons. It is desired to POST COMPUTE average income for display in a third column. The required statements are shown in the next illustration. Although shown in this case, summary values need not be displayed in the table to be included in the calculations specified by the POST COMPUTE statement.

Table A: Region, Income THEN Persons
THEN Average Income;
POST COMPUTE Average Income =
Income / Persons;

	INCOME	PERSONS	AVERAGE INCOME
REGION 1	25,000	5	5,000
REGION 2	30,000	4	7,500
REGION 3	100,000	10	10,000

By setting up logical tests, the SELECT statement allows the user to deal with subsets of information in the data file. The statement itself takes two forms:

SELECT IF (conditions)
or,
SELECT UNLESS (conditions)

Any number of multiple conditions may be separated by AND and OR. A variety of conditions can be tested, including: LESS THAN; GREATER THAN; EQUAL; NOT LESS THAN; NOT GREATER THAN; NOT EQUAL; and GREATER THAN OR EQUAL.

For example, typical expressions:

1. SELECT UNLESS Wages * 52 + Interest IS GREATER THAN INCOME;

2. SELECT IF (a EQUALS 3 OR a IS EQUAL TO 7 OR a=9) AND c < p;

The nomenclature for a given operation permits some freedom of linguistic choice. As examples, the following items within each line are equivalent:

1. EQUAL; EQUAL TO; EQUALS; =; IS EQUAL; IS EQUAL TO.

2. NOT GREATER THAN; LESS THAN OR EQUAL; <=; IS LESS THAN OR EQUAL; IS LESS THAN OR EQUAL TO.

The DEFINE statement allows a new variable to be defined by grouping, deleting, or reordering values of a variable or by all three methods. It takes the form of two columns, one headed by a new variable name and the other by the old variable. For example:

DEFINE

Income group ON Income;
0 through \$49 IF 0:49;
50 through 100 IF 50:100;
All Under 100 IF < 100;
All Other Income IF Other;

"Income group" is the new variable created by grouping the specified values of the old variable "Income." This DEFINE statement is equivalent to:

"If income value is in the range 0 through 49 then the income group to which it belongs is called 0 through \$49" (and so on for the other two categories).

The word OTHER affords unique opportunities to collect into a single class, all incomes not included in any other stipulated category of that DEFINE statement.

Two TPL statements allow median and quantile values to be compiled and displayed. To calculate the median income by region for men and women, the following statements are needed:

MEDIAN Median Income ON Income;

TABLE M1: Region BY Sex, Median Income;

	MEDIAN INCOME
Northeast	
Male	
Female	
Mid-Atlantic	
Male	
Female	
Pacific Coast	
Male	
Female	

The QUANTILE statement is useful, for example, for income percentiles. To uncover the regional income levels below which 25, 50, and 75 percent of the men and women fall, in other words, quartiles, the statement is:

QUANTILE (4) Quartile Income ON Income;

First : 1;

Second: 2;

Third : 3;

The (4) determines the number of quartiles to be calculated. The user selects the ones displayed by citing their corresponding numeric designations; that is, 1, 2, and 3.

This plus the following TABLE statement gives the tabulation displayed below:

TABLE M2: Region BY Sex, Quartile Income;

	QUARTILE INCOME		
	FIRST	SECOND	THIRD
Northeast			
Male			
Female			
Mid-Atlantic			
Male			
Female			
Pacific Coast			
Male			
Female			

The QUANTILE statement (like other TPL statements) can also be used to derive weighted values, where Sample Weight is a part of each data record in the file, as follows below and in table M3:

QUANTILE (4) Weighted Quartiles ON Income

WEIGHTED BY Sample Weight;

First : 1;

Median: 2;

Third : 3;

Tables that computers generate for BLS often deal with time-series data; most of these tables are produced monthly, reflecting the periodicity of the Bureau's major surveys. Although the format of published tables remains the same, reference dates change. Thus, a table showing data for June, July, and August of 1977 and

TABLE M3: Region BY Sex, Weighted Quartiles;

	WEIGHTED QUARTILES		
	FIRST	MEDIAN	THIRD
Northeast			
Male			
Female			
Mid-Atlantic			
Male			
Female			
Pacific Coast			
Male			
Female			

June, July, and August of 1976 for comparison would become July, August, and September 1977 and the same months for 1976 in the succeeding monthly cycle. The relationship among the months remains the same, but the dates change. A TPL statement that expresses this relationship relieves users of the burden of changing date name labels. It is called the RELATIVE TIME statement.

A RELATIVE TIME statement might be used to produce a table with the above dates and median income by region and sex. The first step is to set a reference date with a RELATIVE TIME statement:

RELATIVE TIME Months Ago:

REFERENCE MONTH = August, 1977;

A DEFINE statement is also required as follows:

DEFINE Selected Months ON Months ago;

(YEAR BY MONTH) : LABEL;

: 0;

: 1;

: 2;

: 12;

: 13;

: 14;

TABLE M4: Region BY Sex, Median Income BY Selected Months;

	MEDIAN INCOME					
	1976			1977		
	JUNE	JULY	AUGUST	JUNE	JULY	AUGUST
Northeast						
Male						
Female						
Mid-Atlantic						
Male						
Female						
Pacific Coast						
Male						
Female						

Combining the **DEFINE** and **RELATIVE TIME** statements with **MEDIAN** Median Income **ON** Income; and the **TABLE** statement we get Table M4.

The codebook. The values for the variables specified can be retrieved and processed from the TPL codebook according to the eight TPL statements already discussed. This detailed description must be prepared by someone who knows the file well: Variables must be named; data lengths and locations of values must be specified. Many TPL users use the codebook without knowing the details of file organization.

Preparing a codebook is a one-time activity which the user often performs with the help of a data processing professional.

The ninth TPL statement brings the codebook into play. It is the first one cited in a request for tabulation and must always be included. It takes the form:

USE codebook name **CODEBOOK**;

where "codebook name" is the name assigned to the codebook, e.g.:

USE Labor Force **CODEBOOK**;

When data are maintained in the BLS data base management system, individual codebooks are prepared for each file involved. An "associative" codebook is compiled to describe the relationships between the individual files. Thus, one tabulation may draw on data from multiple files. The **USE** statement then takes the form:

USE associative codebook name **VIEW** number; where "VIEW number" guides the process to the specific set of individual codebooks that describe the data needed.

Functions. Square root and absolute value functions are available plus the ordinary arithmetic functions of addition, subtraction, division, and multiplication. In addition, maximum and minimum functions may be used. **MAX** is an operator that displays the largest value of the variables and must be used with a **POST COMPUTE** statement. For example, the first statement for a table showing the highest income earned by men and women among regions would be:

POST COMPUTE Highest Income=**MAX** (Income);

The **TABLE** statement would be:

TABLE M5: Region BY Sex, Highest Income;

	HIGHEST INCOME
Northeast	
Male	
Female	
Mid-Atlantic	
Male	
Female	
Pacific Coast	
Male	
Female	

In a similar fashion, MIN is an operator (also used with a POST COMPUTE statement) that yields the smallest value of the variable. By replacing MAX in POST COMPUTE with MIN and using the label Lowest, a table would be derived with minimum income by region and sex.

Additional features. TPL also serves table production in other ways. Tabulation results are needed on a clean copy for photo-offset printing in periodicals and bulletins. For this purpose, numbers must be surrounded with a clean and concise framework of explanatory alphabetic information, such as table and column headings and stubs as well as footnotes and similar data that make sense and are readable.

The user of TPL has optional control over the appearance of the table. If he does not exercise this option, the system will automatically format table features such as stubs, columns, and headings using names of variables shown in the codebook. These may not be acceptable as published titles. If the user chooses to control the appearance of the table, then the column and stub widths may be set as desired, and one's own choice of alphabetical labels for each variable table title is permitted. Special features allow hyphenation, and

centering of table titles and column headings. Combined, these features can create tables that are acceptable for direct photo-offset printing of many of the Bureau's tables.

Even though tables for direct photo-offset printing has been made easier, the print facility has been extended one more step. Photo-offset printing of computer printout is less satisfactory than tables composed by special devices, such as the Government Printing Office's electronic photo-composition machines. Built-in controls allow TPL tables that are formed by a photocomposer to have a wide range of print size, style, and other typographic choices. Through this approach, tables appear to be typeset without the expense and need to write a computer program for each table format as occurred before the extension of TPL.

The basic TPL arithmetic calculations were not designed to do many of the complex, scientific analyses required by the Bureau's statisticians, economists, demographers and other analysts.* Researchers may shunt tabulated results into packaged collections of statistical analysis routines or may steer their TPL results into routines of their own making.

Practical Uses of TPL

Except for hardware restrictions, TPL is free of ordinary computer constraints. The many variables that can be tabulated may be nested and concatenated, and results displayed with great freedom. There is no theoretical limit on table dimensions and a large number of tables can be processed in a single run.

Processing hierarchical data files is a featured advantage. In fact, a principal impetus to create the system came from the need to tabulate a major BLS survey characterized by complex hierarchical files.

Some background information on this survey is worth noting. Business enterprises, social scientists, and the Federal Government want to learn how families spend their money. Because the BLS needs this information to compute the monthly CPI, every 10 years it conducts a Consumer Expenditure Survey of 15,000 to 20,000 sample households to determine the contents and costs of the family market basket. During the survey period, expenditures for approximately 6,000 items—the lowest level of a hierarchy of information—range from toothpaste to automobiles, packaged flour to homes, and so forth. There is also interest in expenditures by family member, by the family treated as a unit, by region, and so forth.

Research analysts often are interested in categories of expenditures not identified in advance. It is not always practical to code a collection of family expenditures so that classes defined by all possible combinations and permutations are noted. For a survey of 6,000 variables, the number of possible classes would be astronomical, and meaningful expenditure combinations so large as to make coding impractical.

For example, some researchers may be interested in expenditures for leather (footballs, shoes, belts, etc.), or expenditures for recreational equipment (footballs, baseballs, tennis racquets, bicycles, etc.), or metal products (bicycles, automobiles, refrigerators, etc.). Such a grouping may be desirable for one person, but would have no use for another. Short of precoding all classes, one can see no practical way to anticipate all meaningful combinations. The user of TPL can define groupings of expenditures into classes without limit and still remain inside the language structure.

Another feature without precedent among general-purpose tabulation systems is the ability of TPL, in tabulating from a hierarchical arrangement, to count the number of families making expenditures for a particular item or collection of items, say clothing. A

family of four might have a daughter who bought a suit, a son who bought a sweater, and a parent who purchased a coat. Without the need for other machine instructions, the user of TPL can count the family just once as a unit that purchased clothing, no matter how many family members made purchases and no matter how many items of clothes each purchased.

TPL can provide simultaneous counts at intermediate levels in the family hierarchy. For example, one could count the number of children within the family (one step down the hierarchy) for whom items of apparel were purchased. And the hierarchy can be extended upward and downward: Count (1) the number of geographic areas with families whose members purchased clothing, and (2) the quarter of the calendar year in which the purchases were made. In other words, given a hierarchy of information, the user of TPL can count at any level.

In a similar way, the user of TPL may summarize quantities in addition to just counting occurrences throughout the hierarchy. For example, for families that purchased apparel, a researcher may want to calculate average expenditures for clothing and other apparel (summary of family apparel purchases divided by number of families) and the average expenditures for children's items of apparel among families making purchases of the children's apparel.

The following TABLE and POST COMPUTE statements illustrate the flexibility and range of computational capabilities TPL has for complex files. Apparel, in the Bureau's Consumer Expenditure Survey, is not shown as a class but rather as items of expenditure on shoes, shirts, ties, and so forth. Therefore, we must begin the tabulation of apparel with a DEFINE statement to arrive at a class of expenditures called apparel. Such a statement is simply a listing of the items to be included in the category apparel, somewhat as follows:

```
DEFINE Apparel ON Item;  
    Shoes;  
    Shirts;  
    Dresses;  
    Coats;  
    Etc;
```

Now, these apparel expenditures may be tabulated at two levels in the family hierarchy: By a high level, namely the family unit; and at the intermediate level of purchases for children in the family. The file shows each item of apparel bought as an item of expenditure for the person in the family for whom it was purchased. Therefore, in counting apparel expenditures for the family unit, TPL includes a count of one for each expenditure of apparel, but only once for each family, no matter how many items were purchased. In the meantime, it combines all the purchases for all family members to arrive at total family expenditures. At the same time, the system can keep track of children (a lower hierarchical level) and purchases for them to arrive at expenditures for children. The following TABLE and COMPUTE statements are about what is needed for the table "Counts and Average Apparel Expenditures for Families and Children" shown subsequently.

TOTAL THEN Region BY (TOTAL THEN City),
TOTAL THEN Sex of Head THEN Family Size,
Apparel BY (Number of Families THEN Average
Family Apparel Expenditures THEN Number of
Children THEN Average Children's Apparel Expend-
itures;

POST COMPUTE

Average Family Apparel Expenditures = Total Family
Apparel Expenditures/Number of Families;

POST COMPUTE

Average Children's Apparel Expenditures = Total
Children Apparel Expenditures/Number of Children;

SEATTLE					
PACIFIC REGION					
BOSTON		COUNTS AND AVERAGE APPAREL EXPENDITURES FOR FAMILIES AND CHILDREN			
NEW ENGLAND REGION		COUNTS AND AVERAGE APPAREL EXPENDITURES FOR FAMILIES AND CHILDREN			
UNITED STATES		COUNTS AND AVERAGE APPAREL EXPENDITURES FOR FAMILIES AND CHILDREN			
		NUMBER OF FAMILIES	AVERAGE FAMILY APPAREL EXPENDITURES	NUMBER OF CHILDREN	AVERAGE CHILDREN APPAREL EXPENDITURES
	TOTAL				
	SEX OF HEAD OF FAMILY				
	MALE				
	FEMALE				
	FAMILY SIZE				
	ONE				
	TWO				
	THREE				
	FOUR				
	FIVE				

All operations producing the table from data within the hierarchy are unique since they are performed directly by TPL. The user need not provide procedural or machine language subprogramming, preprocess the file or do similar activities outside TPL, nor rely on computer specialists to do the work. The machine translates user statements into procedures it must

follow to get the product. From the user's viewpoint, TPL is problem oriented, rather than procedure or process oriented as are most computer languages. In other words, TPL (the computer system) already knows what a table is and how to generate one. It needs only to be told the particulars about the one wanted.

Technical Considerations

The many computer users today form a big market for vendors of computer software. The market is not monolithic, but is divided in ways that compel software manufacturers to devise tools and sales strategies that vary according to the target population. For example, software requirements for a bank differ from the needs of a manufacturing plant or an airline. Nevertheless, vendors are creating a wide range of useful software tools for sale or lease to various segments of the customer population because each portion is large enough to make the venture profitable. As a result, many enterprises find that their software needs are increasingly served by proprietary products. In the commercial world, general programs for payroll processing, inventory control, and other business data processing requirements are fairly common.

Special problems at national statistical agencies. Data processing problems in the BLS reflect conditions in other national statistical agencies—conditions not ordinarily found in commercial ventures. Consequently, software vendors have not done much to help such agencies because the market is small and scattered around the globe.

Photocomposition is a standard process in the publication business and a good deal of money has been invested in tying the computer into printing technology. This partnership has supported the design and development of sophisticated software that takes alphabetic characters and photocomposes text for newspapers and other large volume publications. Statistical agencies produce relatively little text. As an instance, the BLS conducts decennial studies of family expenditures for goods and services. One such effort resulted in printing statistical tables, with almost no text, in 18 fat volumes.

Had the general community of computer users printed massive amounts of statistical data in tabular format, software houses would have been attracted by the size of the market and filled the demand for software to photocompose statistical tables. However,

national statistical agencies are not beyond help. Several years ago, the BLS acquired a data base management program from a software vendor. As a result, special data management programs need not be written for most Bureau survey systems. The Bureau also benefits from work pioneered in universities where highly sophisticated but readily used packages for statistical analysis have been developed. BLS uses several of these: SAS, North Carolina State University; SOUPAC, a package of routines from the University of Illinois; and SPSS, University of Chicago. In summary, although vendors of software can supply generalized data management routines and the need for analytical routines is met by university products, the Bureau's unique requirements for tabulation and publication of statistical tables have not been met.

TPL compared. Interest in, and concern about, the unique data processing problems associated with cross tabulation of variables in large-scale files, such as population censuses, is not confined to national statistical agencies.¹⁰ Owing to widespread interest, the American Statistical Association (ASA) compared twelve prominent systems.

The rating, performed by the ASA Committee on the Evaluation of Statistical Program Packages,¹¹ showed TPL at the top on all nine criteria used, a distinction not given to any of the other eleven software packages.

The nine ASA evaluation criteria for rating packages were grouped under two general categories: tabulating power and simplicity of language. Tabulating power included the following four criteria: (1) arithmetic ability, (2) placement of numbers on the page, (3) labeling ability, and (4) visual impact. Simplicity of language included the following five criteria: (1) English-like syntax, (2) English-like symbols, (3) consistent semantics, (4) concise specifications, and (5) convenient programming features. Using a scale from 1 (poor) to 5 (excellent) the Committee gave TPL a score of 5 on all counts, 45 altogether. The nearest competitor to TPL was GENSTAT, developed at the Rothamsted Experimental Station, Harpenden, England, scoring 36 of the 45 points possible.

—FOOTNOTES—

¹Rudolph C. Mendelssohn, "The Principles of Processing Statistical Data," *The American Statistician*, Vol. 24, October 1970.

²Hugh F. Brophy, "The Requirements of a Generalized Report Generator Language," *Proceedings of the Fourth Australian Computer Conference*, Adelaide, South Australia, 1969.

³*Table Producing Language (TPL) USERS Guide*, Version 3.5, February 1975 (Bureau of Labor Statistics, Office of Systems and Standards).

⁴Barry W. Smith, "Symbolic Notations for Statistical Tables and An Approach Automatic Systems Design," *Communications of The Association for Computing Machinery*, Vol. 8, No. 6, June 1965.

⁵William M. McKeeman, James J. Horning and David B. Wortman, *A Compiler Generator*, Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1970.

⁶BNF-Bachus-Naur form, a metalanguage for describing the grammatical structure of languages.

⁷Peter B. Stevens, *The BLS Information System, a Systems Manual*, Bureau of Labor Statistics, Washington, D.C., September 1967.

⁸L. Ion, "Generalized Table Generator," *Proceedings of the Fourth Australian Computer Conference*, Adelaide, South Australia, 1969.

⁹Rudolph C. Mendelssohn, "Data Banks and Information Systems for National Statistics," *Proceedings: Congress of the International Federation for Information Processing*, Ljubljana, Yugoslavia, August 1971.

¹⁰Gary L. Hill, Lawrence L. Brown, Kisun Han, "Maximizing Access to Public Use Samples," *Review of Public Data Use*, Vol. 1, No. 1, December 1972.

¹¹Ivor Francis, Stephen P. Sherman, Richard M. Heiberger, "Languages and Programs For Tabulating Data From Surveys," *Proceedings of Computer Science and Statistics Ninth Annual Symposium on the Interface*, Cambridge, Massachusetts, 1976.

How to order BLS publications

MONTHLY LABOR REVIEW

Order from Monthly Labor Review, Box 353, La Plata, Md. 20646. Make checks payable to Superintendent of Documents.

The oldest and most authoritative government research journal in economics and the social sciences. Current statistics, analysis, developments in industrial relations, court decisions, book reviews. \$16 a year, single copy \$1.40.

OTHER PERIODICALS

Order from (and make checks payable to) Superintendent of Documents, Washington, D. C. 20402. For foreign subscriptions, add 25 percent.

Employment and Earnings. A comprehensive monthly report on employment, hours, earnings, and labor turnover by industry, area, occupation, etc. \$18 a year, single copy \$1.50.

Occupational Outlook Quarterly. A popular periodical designed to help high school students and guidance counselors assess career opportunities. \$4 for four issues, single copy \$1.30.

Current Wage Developments. A monthly report about collective bargaining settlements and unilateral management decisions about wages and benefits; statistical summaries. \$12 a year, single copy \$1.35.

Wholesale Prices and Price Indexes. A comprehensive monthly report on price movements of both farm and industrial commodities, by industry and stage of processing. \$16 a year, single copy \$1.80.

CPI Detailed Report. A monthly periodical featuring detailed data and charts on the Consumer Price Index. \$9 a year, single copy \$0.75.

Chartbook on Prices, Wages, and Productivity. Monthly, presents 19 analytical charts and detailed supporting tables on price, wage, and productivity movements. \$11 a year, single copy \$0.95.

PRESS RELEASES

The Bureau's statistical series are made available to news media through press releases issued in Washington. Many of the releases also are available to the public upon request. Write: Bureau of Labor Statistics, Washington, D.C. 20212.

BULLETINS AND HANDBOOKS

About 140 bulletins and handbooks published each year are for sale by regional offices of the Bureau of Labor Statistics (see inside front cover) and by the Superintendent of Documents, Washington, D. C. 20402. Make checks payable to the Superintendent of Documents. Among the bulletins and handbooks currently in print are these:

Handbook of Labor Statistics 1977. Bulletin 1966. A 361-page volume of historical data on the major BLS statistical series. \$5.50.

Handbook of Methods. Bulletin 1910. Brief technical account of each major statistical program of the Bureau of Labor Statistics. \$3.50.

BLS Measures of Compensation. Bulletin 1941. An introduction to the various measures of employee compensation; describes each series, the manner in which it is developed, its uses and limitations. \$2.75.

Productivity and the Economy. Bulletin 1926. A colorful chartbook that shows what productivity is and how it interacts with other aspects of the economy. \$2.75.

Brief History of the American Labor Movement. Bulletin 1000. Describes the development of the U.S. trade unions. \$1.45.

Directory of National Unions and Employee Associations, 1975. Bulletin 1937. National and international unions, State labor organizations, professional and public employee associations are listed. \$2.75.

U. S. Workers and Their Jobs: The Changing Picture. Bulletin 1919. Presents graphically data from the major BLS statistical series. \$0.60.

REPORTS AND PAMPHLETS

Single copies available free from the BLS regional offices or from the Bureau of Labor Statistics, Washington, D.C. 20212.

Revising the Consumer Price Index. A nontechnical explanation of what is involved in revising the Consumer Price Index.

Employment and Unemployment in 1976. An analysis of employment and joblessness during the past year.

How the Government Measures Unemployment. Report 505. A concise report providing a background for appraising developments in the area of unemployment.

Publications of the Bureau of Labor Statistics, July-December 1976. Report 496. A listing, with selected annotations, of all BLS publications from July through December 1976.

Major Programs 1977, Bureau of Labor Statistics. Report 488. A 54-page pamphlet explaining the scope of the Bureau's principal programs.

Regional Publications. Each of the Bureau's eight regional offices publishes reports and press releases dealing with regional data. Single copies available free from the issuing regional office.

END

2-3-82